

(Optional) Two-Factor Authentication (2FA) : Part 2

Real World Application ([index.html](#))

- Real World Application
 - index.html
 - Status Check Section
 - Setup 2FA Section (setup.php)
 - Reset Check Section (reset.php)
 - Test 2FA Login Section (login.php)
 - Verify Section

Real World Application

```
cd DIRECTORY/two_factor  
php -S localhost:8000
```

Open a web browser and access <http://localhost:8000/>

index.html

- Status Check Section
- Setup 2FA Section
- Reset Section
- Test Login Section
- Verify Section

Program Pattern

1. From the HTML section, read users' input; it maps to a corresponding JavaScript.
 - i. In this example, checkStatus is linked to this HTML form via a button.

```
<div class="button-group">  
  <button class="status-btn" onclick="checkStatus(false)">Check Basic Status</button>  
  <button class="status-btn" onclick="checkStatus(true)">Check Detailed Status (with password)</button>  
</div>
```

2. In JavaScript, it makes a POST request to the server.
 - i. In this example, make a POST request to `setup.php`.

```
const response = await fetch('setup.php', {  
  method: 'POST',  
  headers: { 'Content-Type': 'application/json' },  
  body: JSON.stringify(requestBody)  
});
```

3. In JavaScript, get the response and update the HTML placeholders.

```
if (response.ok) {  
  responseDiv.className = 'response success';  
  responseDiv.innerHTML = `✅ 2FA Setup ${forceReset ? '(After Reset)' : 'Started'}!\n\n${JSON.stringify(data, null, 2)}`;   
  
  document.getElementById('qrCodeImage').src = data.qr_code_url;  
  document.getElementById('manualKey').textContent = data.manual_entry_key;  
  document.getElementById('qrCodeContainer').style.display = 'block';  
  document.getElementById('continueToVerify').disabled = false;  
}
```

How to read the code

1. Understand the index.html first.
 - i. Understand the HTML form (HTML)
 - ii. Understand the JavaScript (JavaScript functions)
 - iii. Understand what request is made and what response to get (Request and Response)
 - iv. Understand how to update the HTML placeholders from the response (Update)
2. Understand the PHP server programs in later sections.

How many sections are in the index.html?

*There are five sections and their endpoints on the PHP server

- Status Check Section (status.php)
- Setup 2FA Section (setup.php)
- Reset Section (reset.php)
- Test Login Section (login.php)
- Verify Section (verify.php)

Status Check Section

HTML

- There are two demo users (radiobutton) and two buttons in this section.

```
<div>
  <label><input type="radio" name="statusUser" value="john" checked> john</label>
  <label><input type="radio" name="statusUser" value="admin"> admin</label>
</div>
```

- Click the button to call the checkStatus JavaScript function.

```
<div class="button-group">
  <button class="status-btn" onclick="checkStatus(false)">Check Basic Status</button>
  <button class="status-btn" onclick="checkStatus(true)">Check Detailed Status (with password)</button>
</div>
```

JavaScript

getSelectedUser()

- Find all the radio buttons with the "input name =" attribute.
- Return if the radio button is checked for the user.

```
function getSelectedUser(radioName = 'user') {  
  const radios = document.querySelectorAll(`input[name="${radioName}"]`);  
  for (let radio of radios) {  
    if (radio.checked) {  
      return radio.value;  
    }  
  }  
  return 'john';  
}
```

checkStatus()

- From the radiobutton, get the selected user.
- Get the password and find the placeholders.

```
const user = getSelectedUser('statusUser');  
const password = user === 'john' ? 'password123' : 'admin123';  
const responseDiv = document.getElementById('statusResponse');  
const statusDiv = document.getElementById('statusDisplay');
```

Request & Response (status.php)


- From the `status.php`, get the user information.

```
let response;
if (detailed) {
  response = await fetch('status.php', {
    method: 'POST',
    headers: { 'Content-Type': 'application/json' },
    body: JSON.stringify({ username: user, password })
  });
} else {
  response = await fetch(`status.php?username=${user}`);
}

const data = await response.json();
```

Update

- Update the HTML out of the response from `status.php`


```
responseDiv.className = 'response info';
responseDiv.innerHTML = ` Status Check Result:\n\n${JSON.stringify(data, null, 2)}`;

// Show user-friendly status
let statusHtml = `

### Status for ${user}:</h3>`; statusHtml += ` <strong>2FA Enabled:</strong> ${data.totp_enabled ? '✅ Yes' : '❌ No'}</p>`; statusHtml += ` <strong>Has Secret:</strong> ${data.has_secret ? '✅ Yes' : '❌ No'}</p>`;


```

- Make statusHtml string and add information until statusDiv.innerHTML is set to `statusDiv.innerHTML = statusHtml; .`
- Make the statusDiv visible with `statusDiv.style.display = 'block';`

```
if (data.setup_status) {
  statusHtml += `<p><strong>Setup Status:</strong></p><ul>`;
  for (let [key, value] of Object.entries(data.setup_status)) {
    if (value) {
      statusHtml += `<li>${key.replace(/_/g, ' ')}: </li>`;
    }
  }
  statusHtml += `</ul>`;
}

if (data.available_actions) {
  statusHtml += `<p><strong>Available Actions:</strong> ${data.available_actions.join(', ')}</p>`;
}
```

Setup 2FA Section (setup.php)

- Start 2FA setup

HTML

```
<div>
  <label><input type="radio" name="user" value="john" checked> john (password: password123)</label><br>
  <label><input type="radio" name="user" value="admin"> admin (password: admin123)</label>
</div>
<div class="button-group">
  <button onclick="startSetup(false)">Start Normal Setup</button>
  <button onclick="startSetup(true)">Force Reset & Setup</button>
</div>
<div id="setupResponse" class="response" style="display:none;"></div>
```

JavaScript

startSetup(forceReset = false)

- Get user and set force_reset property.

```
selectedUser = getSelectedUser();
```

Reset Check Section (reset.php)

Two reset functions (reset2FA() and forceResetAndSetup()) serve related but distinct purposes in the 2FA management workflow:

`reset2FA()` Function

Purpose: Only resets/disables 2FA

- Gets the user from the `resetUser` radio buttons
- Sends a POST request to `reset.php` with `confirm_reset: true`
- **Only disables** the 2FA for the selected user
- Displays the result in the reset response area
- **Does not** automatically start a new setup

`forceResetAndSetup()` Function

Purpose: Resets 2FA AND immediately starts new setup

- Gets the user from the `resetUser` radio buttons
- **Automatically sets** the setup radio button to match the selected user
- Calls `startSetup(true)` which triggers the setup process with force reset
- **Combines** reset and setup in one seamless action

Key Differences

Aspect	<code>reset2FA()</code>	<code>forceResetAndSetup()</code>
Action	Reset only	Reset + Setup
API Calls	<code>reset.php</code>	<code>setup.php</code> (with <code>force_reset</code>)
User Flow	Manual next step	Automatic continuation
Use Case	"I want to disable 2FA"	"I want to replace current 2FA"

Think of it like **resetting a phone**:

- `reset2FA()` = Factory reset → phone is wiped, but you need to start setup manually
- `forceResetAndSetup()` = Factory reset + auto-boot into setup wizard → one-click solution

HTML

- We can reset with the `reset2FA()` and `forceResetAndSetup()` .

```
<div>
  <label><input type="radio" name="resetUser" value="john" checked> john (password: password123)</label>
  <label><input type="radio" name="resetUser" value="admin"> admin (password: admin123)</label>
</div>
<div class="button-group">
  <button class="reset-btn" onclick="reset2FA()">Reset 2FA</button>
  <button onclick="forceResetAndSetup()">Reset & Start New Setup</button>
</div>
<div id="resetResponse" class="response" style="display:none;"></div>
```

JavaScript

reset2FA()

- Get the user to reset their password.
- Make sure that the user wants to reset the 2FA.

```
const user = getSelectedUser('resetUser');
const password = user === 'john' ? 'password123' : 'admin123';
const responseDiv = document.getElementById('resetResponse');

if (!confirm(`Are you sure you want to reset 2FA for ${user}? This will disable 2FA and require e-setup.`)) {
    return;
}
```

Request & Response (reset.php) and Update

```
const response = await fetch('reset.php', {
  method: 'POST',
  headers: { 'Content-Type': 'application/json' },
  body: JSON.stringify({ username: user, password, confirm_reset: true })
});

const data = await response.json();

if (response.ok) {
  responseDiv.className = 'response success';
  responseDiv.innerHTML = `✅ 2FA Reset Successful!\n\n${JSON.stringify(data, null, 2)}`;
} else {
  responseDiv.className = 'response error';
  responseDiv.innerHTML = `❌ Reset Failed:\n\n${JSON.stringify(data, null, 2)}`;
}
```

forceResetAndSetup()

- Compared to reset2FA(), forceResetAndSetup() uses startSetup() function to initialize the user.

```
const user = getSelectedUser('resetUser');

if (!confirm(`Reset 2FA for ${user} and start new setup immediately?`)) {
    return;
}

// Set the user radio button for setup
document.querySelector(`input[name="user"][value="${user}"]`).checked = true;

// Start setup with force reset
await startSetup(true);
```

Test 2FA Login Section (login.php)

HTML

```
<div class="step" id="step4">
  <h2>Step 4: Test 2FA Login</h2>
  <p>Now test the complete login process with 2FA:</p>
  <input type="text" id="loginCode" placeholder="000000" class="code-input" maxlength="6">
  <button onclick="testLogin()">Login with 2FA</button>
  <div id="loginResponse" class="response" style="display:none;"></div>
</div>
```

JavaScript

testLogin()

- Make sure the setup is complete.
- The login code should be correctly provided.

```
if (!setupComplete) { alert('Please complete the 2FA setup first!'); return; }

const code = document.getElementById('loginCode').value;
```

- Add a listener to "verifyCode" and "loginCode" placeholders to allow only numeric input.

```
// Allow only numeric input in code fields
document.getElementById('verifyCode').addEventListener('input', function(e) {
    this.value = this.value.replace(/\D/g, '');
});

document.getElementById('loginCode').addEventListener('input', function(e) {
    this.value = this.value.replace(/\D/g, '');
});
```


Request & Response

- Make a POST request with a JSON string.

```
const response = await fetch('login.php', {
  method: 'POST',
  headers: { 'Content-Type': 'application/json' },
  body: JSON.stringify({
    username: selectedUser,
    password,
    totp_code: code
  })
});

const data = await response.json();
```

Update

```
if (response.ok) {
  responseDiv.className = 'response success';
  responseDiv.innerHTML = `🚀 2FA Login Successful!\n\n${JSON.stringify(data, null, 2)}`;
  document.getElementById('step4').classList.add('completed');
}
```

Verify Section

- Verify 2FA setup

HTML

```
<div class="step" id="step3">
  <h2>Step 3: Verify Setup</h2>
  <p>Enter the 6-digit code from your authenticator app:</p>
  <input type="text" id="verifyCode" placeholder="000000" class="code-input" maxlength="6">
  <button onclick="verifySetup()">Verify & Enable 2FA</button>
  <div id="verifyResponse" class="response" style="display:none;"></div>
</div>
```

JavaScript

verifySetup()

```
const code = document.getElementById('verifyCode').value;
const responseDiv = document.getElementById('verifyResponse');

if (!code || code.length !== 6) {
```